REMARKS/ARGUMENTS

These remarks are submitted in response to the Office Action dated March 21, 2006 (Office Action). As this response is timely filed before the expiration of the 3-month shortened statutory period, no fee is believed due.

Claims 1, 5, 6, 9, 10, 12, 14, 17, 18, 20, 24, 25, 28, 29, 33, 36, and 37 were rejected at page 3 of the Office Action under 35 U.S.C. § 102(e) as being anticipated by U.S. Published Patent Application No. 2002/002313 to Wu, *et al.* (Wu). Claims 3, 4, 13, 16, 22, 23, 32, and 35 were rejected at page 5 of the Office Action under 35 U.S.C. § 103(a) as being unpatentable over Wu in view of U.S. Patent Number 6,697,474 to Hanson, *et al.* (Hanson). Claims 2, 15, 21, and 34 were rejected at page 6 of the Office Action under 35 U.S.C. § 103(a) as being unpatentable over Wu in view of U.S. Patent No. 6,757,365 to Bogard (Bogard).

Applicants have amended each of independent Claims 1, 9, 18, 20, and 28 to emphasize certain aspects of Applicants' invention. The amended claims are fully supported throughout the Specification, as discussed herein. (See, e.g., Specification, pp. 3-4, line 27 - line 2; p. 10, lines 9-19; and p. 11, lines 6-11.) No new matter has been introduced by virtue of these amendments.

Applicants' Invention

It may be useful to reiterate certain aspects of Applicants' invention prior to addressing the cited references. One embodiment of the invention, as typified by amended independent Claim 1, is an instant messaging or chat (IM/chat) communication method. The method can include inserting in a program code of an instant message (IM)

a) a header component that provides identity information of an IM sender in a voice conference identifier for connecting a voice communication link from a recipient client to the IM sender, b) a body component that provides message text that is displayed prior the connecting a voice communication link, and c) an attachment component for including the voice conference identifier comprising a voice conference call list identifying conference call nodes. The method also can include embedding compiled object code of the voice conference identifier in the attachment. (See, e.g., Specification, pp. 10, line 15 - line 19) Execution of the computer compiled object code at a recipient node establishes a voice communications link between the IM sender and a recipient at the recipient node. (See, e.g., Specification, p. 11, lines 6-11.)

The method further can include transmitting the IM from the sender to the recipient at the recipient node. At the recipient node, a recipient client can present the identity information with the message text and detect the voice conference identifier within the IM. Responsive to the detecting step, an IM interface of the recipient can display a user-selectable text or graphic symbol, and responsive to a user-selection of the displayed text or graphic symbol, a voice communication link between the recipient and the sender can be established by executing the compiled object code embedded in the IM on a machine hosting the IM interface.

The Claims Define Over The Prior Art

As already noted, independent Claims 1, 9, 18, 20, and 28 were rejected as being anticipated by Wu. Applicants respectfully submit that Wu fails to expressly or inherently teach every aspect of Applicants' invention. None of the references, alone or in combination, teach or suggest every feature of Applicants' invention.

For example, none of the references teach the embedding of compiled object code in an IM, as recited in amended independent Claims 1, 9, 18, 20, and 28. Accordingly, Wu and other the references further fail to teach the establishment of a voice communications link between an IM sender and a recipient by executing computer compiled object embedded in an IM, as recited in each of independent Claims 1, 9, 18, 20, and 28, as amended. The references likewise fail to teach that the compiled object is executed at a recipient node or that it is executed in response to a recipient's selecting a symbol displayed in response to detecting the IM at the recipient node, as also recited in the amended independent claims.

Wu concerns the transferring of electronic data between a sender and a recipient. Electronic data transfer in Wu includes enabling instant messaging communications between the sender and at least one recipient. (See, e.g., paragraphs [0005] and [0014]; see also Abstract.) Notably, instant messaging in Wu is effected exclusively through "an instant messaging host" or host system. More particularly, both the sender and recipient in Wu communicate with one another by connecting to the host system. The host system authenticates text messages between the sender and recipient, determines and reports the capabilities of each, and, most importantly in the present context, establishes a talk session between the sender and recipient.

Wu, however, nowhere relies on sending an instant message having embedded compiled object code for establishing a voice communication link between the sender and recipient. In Wu, it is the host, to which both the sender and recipient connect, that establishes a voice communication link. As clearly shown in FIG. 6, the sender **602a** in Wu connects to a host **604**. The recipient **602b** in Wu likewise connects to the host **604**. To establish a talk session, the host **604** in Wu authenticates a talk request from the sender **602a** and sends the request to the recipient **602b**. No software for establishing the talk session is embedded in the forwarded request to the recipient **602b**. The software for establishing the talk session must reside at the host **604** because it is explicitly the host that carries out the steps for establishing the talk session. (Wu, Paragraphs 0072-0075 and, especially, FIG. 6, steps **655**, **660**, **665**, and **670**.)

Accordingly, Wu is distinct from Applicants' invention in at least two significant respects. Firstly, Wu does not expressly or inherently teach the sending of an IM message that includes embedded software for establishing a voice communications link. Secondly, the software for establishing the voice communications link is not executed at a recipient node; in Wu the software establishing the voice communications link (i.e., the "talk session") executes in a host system to which both a sender and recipient connect.

In rejection of Claim 1 Paragraph 9 on page 3 of the Office Action, the Examiner points out that Wu establishes the voice communication link upon selecting the graphical symbol via embedded computer program code (See Wu [0071-0074]). Applicant's respectfully agree with the Examiner that the START TALK UI button described in paragraphs [0066-0074] is a software controlled button, and, upon selection, "initiates a talk session by sending a talk request to the host" (See Wu [0071]). However, the talk button is most likely activated via a scripting language which is clearly different than executing compiled object code on the local machine. Wu does not disclose the operations of the START TALK UI button, though it is clear that the action sends a message to a host for initiating a voice chat. Applicants do not employ a host, nor a messaging system for activating a UI. Moreover, Applicant's clearly state that the program component in the IM message is a compiled object that can be independently executed without requiring intermediate script interpretation (See Applicants Specification, page 10, lines 15-25). The program component of Applicant's invention is generated in response to an execution of compiled code on a local machine. The software controlled button of Wu's invention is most likely a scripting action.

It is common for internet browsers that provide chat or message service to provide scripting language interpreters for providing graphical user interface control. Software control actions are processed in the browser and communicated to a host server. For example, the host server can provide call setup for establishing a voice chat between two parties. Notably, the host, which is generally a remote server, may coordinate the call set up and controls communication for supporting the voice chat. That is, the host serves as a moderator for the voice chat. Wu clearly employs a host for providing voice chat operation (See Wu [0071-0072] and Figures 2-6). In contrast, Applicant's invention does not employ a host, and performs the voice communication processes on a local machine.

Applicant's invention runs a program directly on a machine hosting the IM interpreter. That is, upon selection of the voice conference identifier to initiate a voice call, a compiled code object is executed on the local machine to run a program that establishes a voice communication action. For example, referring to FIG. 2 of Applicant's invention, the method call "CallSender" 56 executes a compiled code object to run a program called CallSender in response to a user selection of the graphical symbol 36. The compiled code object can be included in the IM message 50 as an attachment 50C. Clearly, Applicant's invention is not sending a request to a host, or requesting the host to establish a voice chat on behalf of the clients. It should also be noted, that the program run in response to the executing the compiled object code, called in response to a user selection of the voice conference identifier, establishes the voice communication link; not the host. That is, the program for establishing a voice communication link is executed directly from compiled object code either provided within the IM message, or downloaded from a network (See Applicant's Specification, Pg 10 Lines 15-20). Notably, the attachment component within the IM message can include the complied object for establishing the voice communication link. Wu does not teach embedding compiled object code that is executed on a local machine in response to a user-selection of a graphic symbol or displayed text.

Moreover, program code, or scripting code, is not the same as compiled code. Program code consists of statements for executing a series of instructions. In order to execute the instructions, the program code must be compiled to generate executable code, which is called a compiled object. The compilation of program code first involves interpreting the program code which can consist of parsing the program code, identifying structures, instruction statements, memory arrangements, and linking operations. The compilation of program code generates executable code which can then be run on a processor of a machine. Notably, different machines compile code differently. For example, a Unix machine may compile code using a little-endian format, and a PC desktop may use an Intel format using a big-endian format, wherein an endian formant identifies the byte ordering of the data. Similarly, the compilation depends on the data word sizes. For example, one machine may use 32 bit addresses, whereas a second machine may use 64 bit addresses. Notably, the resultant executable code, or compiled object code, generated from a compilation on a local machine is particular to the machine. That is, the executable code may not run on other machines if the processors are different.

Also, certain programming languages provide for virtual engines which can reside on a machine or can be downloaded to the machine. For example, instead of adding a new piece of hardware, such as a processor, for executing a compiled code object, the virtual machine can provide a translation of the compiled object code to the processor. Accordingly, the compiled code object can be run on a machine that may not have compiled the code, but that does have a virtual machine for running the executable code.

The virtual machine is a program that can be provided as a compiled object in accordance with one embodiment of the invention. The virtual machine can also include program interfaces for establishing voice communication. That is, the program code represented by the compiled object and inserted in the IM message, can establish voice communication links between an IM sender and a recipient client.

One advantage of compiled object code is that it may execute faster on a machine, versus the computational steps of first compiling the code, linking the code, and then executing the code on a server. Executable code (i.e. compiled object code) runs faster than program code. Albeit, a virtual machine running on a processor that provides translation of the executable code does slightly slow down the execution speed. Accordingly, one advantage of embedding an compiled code object, as forwarded by the invention, is to increase a speed of execution. Wu does not teach embedding object code for increasing an execution speed of a program.

Moreover, Applicant's invention causes the compiled object code to be executed at the recipient node, thereby off-loading any processing at the IM sender. Notably, Applicant's invention does not require a server to provide IM voice communication translation. That is, the voice communication between the IM sender and the recipient client can be peer-to-peer. Accordingly, the compiled object code executed at the recipient client provides the majority of processing. Consequently, the IM sender is alleviated from performing core processing tasks associated with maintaining voice communication. Wu nor Hanson disclose or teach embedding compiled object code to relieve processing on a network node and delegate the processing to a local machine. Wu and Hanson are not directed to process handling, and are satisfied with the call-handling performance offered by the packet based networks and servers within the telecommunication networks. Wu nor Hanson teach embedding an object code to relieve processing at a server, nor would there be a motivation for them to do so, as they are directed to using a server for providing voice communication.

In rejection of Claim 2 Paragraph 30, on page 6 of the Office Action, the Examiner points out that Wu fails to disclose the use of VoIP for initiating a voice communication link. However, the Examiner points out that Bogard in the same field of endeavor discloses the use of VoIP for initiating a voice communication link. However, Applicant's invention is directed to executing a compiled object on a local machine in response to a user-selection of a voice conference identifier in a chat application. Bogard discloses that "some embodiments of the invention may allow users with a VoIP phone to access the voice portal" (See Bogard, Col 5, lines 42-54. Bogard is silent as to the embodiments using VoIP. Applicant's respectfully agree that the invention of Wu could be modified with the teachings of Bogard in order to allow an alternate method of voice communication. However, Wu fails to teach every aspect of Applicant's invention as amended in the claims as previously mentioned.

Applicants respectfully maintain, therefore, that Wu fails to expressly or inherently teach every feature recited in amended independent Claims 1, 9, 18, 20, and 28, and that the claims thus define over the prior art. The prior art of Hanson and Bogard also fails to provide a basis for rejecting the dependent claims since each recites yet additional features over those set forth in the independent claims from which each depends. Accordingly, Applicants respectfully maintain that, whereas each of the remaining claims depend from one of the amended independent claims while reciting additional features, dependent Claims 2-7, 10, 11-17, 21-25, and 29-37 likewise define over the prior art.

CONCLUSION

Applicants believe that this application is now in full condition for allowance, which action is respectfully requested. Applicants respectfully request that the Examiner call the undersigned if clarification is needed on any matter within this Amendment, or if the Examiner believes a telephone interview would expedite the prosecution of the subject application to completion.

Respectfully submitted,

Date: **June 19, 2006**

[illegible]

Gregory A. Nelson, Registration No. 30,577
Richard A. Hinson, Registration No. 47,652
Marc A. Boillot, Registration No. 56,164
AKERMAN SENTERFITT
Customer No. 40987
Post Office Box 3188
West Palm Beach, FL 33402-3188
Telephone: (561) 653-5000